

XComment.io - Comments Remover Plugin

This plugin removes comments from source files for a number of programming languages. Required Python 3.6 on Jenkins server. A new build step is added: 'Invoke Comments Remover' which accepts files to process as input and creates uncommented version of them.

User can specify Python path in global settings for the plugin (otherwise the one on the system PATH is used).

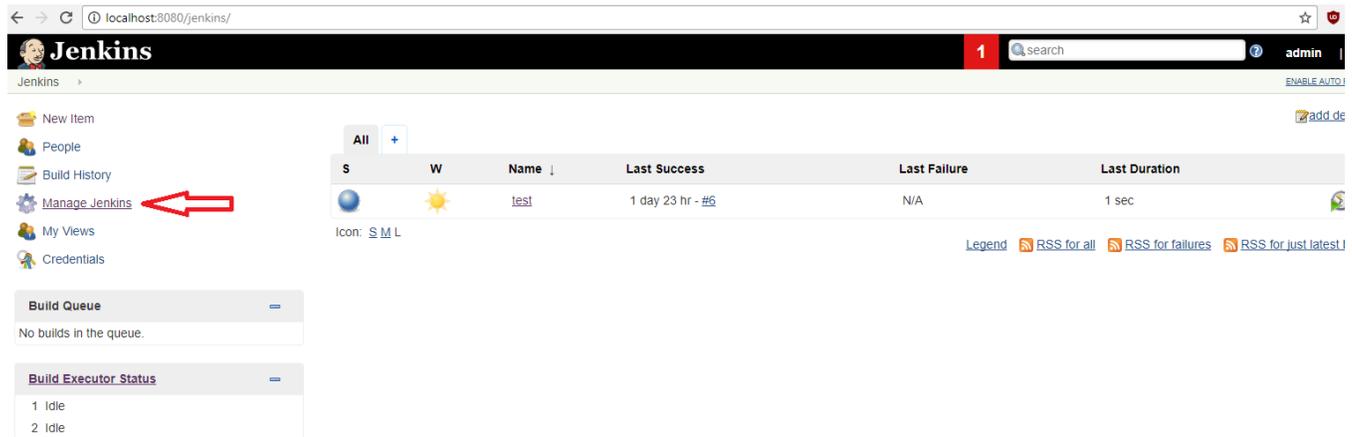
Contributors

Daniel Dylg

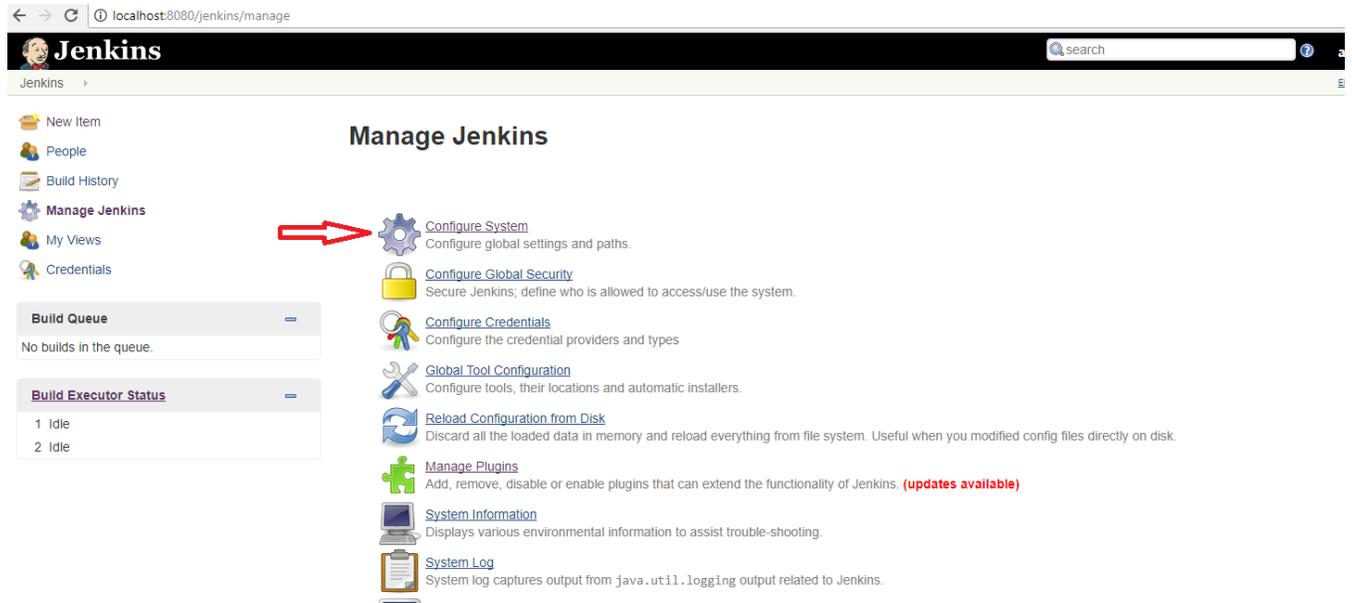
For users

Global settings

Go to Manage Jenkins -> Configure System to access them



The screenshot shows the Jenkins main dashboard at localhost:8080/jenkins/. The left sidebar contains navigation links: New Item, People, Build History, Manage Jenkins (highlighted with a red arrow), My Views, and Credentials. Below the sidebar are sections for Build Queue (No builds in the queue) and Build Executor Status (1 Idle, 2 Idle). The main content area displays a table of builds with columns: S, W, Name, Last Success, Last Failure, and Last Duration. A single build is listed with Name 'test', Last Success '1 day 23 hr - #6', and Last Failure 'N/A'. At the bottom right, there are links for Legend, RSS for all, RSS for failures, and RSS for just latest I.



The screenshot shows the Jenkins 'Manage Jenkins' page at localhost:8080/jenkins/manage. The left sidebar contains navigation links: New Item, People, Build History, Manage Jenkins (highlighted with a red arrow), My Views, and Credentials. Below the sidebar are sections for Build Queue (No builds in the queue) and Build Executor Status (1 Idle, 2 Idle). The main content area is titled 'Manage Jenkins' and lists several configuration options, each with an icon and a brief description: Configure System (gear icon), Configure Global Security (lock icon), Configure Credentials (key icon), Global Tool Configuration (wrench icon), Reload Configuration from Disk (refresh icon), Manage Plugins (puzzle piece icon), System Information (laptop icon), and System Log (notepad icon). A red arrow points to the 'Configure System' option.

Comments Remover

Python path



Set this field if you don't want to use the default Python executable (available as 'python' on system path)

Pip path



Set this field if you don't want to use the default pip executable (available as 'pip' on system path)

Verbose mode



If true whole process output will be printed to the console log of a build

There is help section to provide examples:

Comments Remover

Python path



Set this field if you don't want to use the default Python executable (available as 'python' on system path)

Examples:

/usr/bin/python3

C:\venv\comments_remover_venv\Scripts\python.exe

(from [Comments Remover plugin](#))

Pip path



Set this field if you don't want to use the default pip executable (available as 'pip' on system path)

Examples:

/usr/bin/pip2

C:\venv\comments_remover_venv\Scripts\pip.exe

(from [Comments Remover plugin](#))

Verbose mode



If true whole process output will be printed to the console log of a build

This options will cause output stream and error stream of comments_remover process to be redirected, so that the user can see the process of running it in the build console log.

(from [Comments Remover plugin](#))

Usage

The plugin can be used wherever build steps can be defined, e.g. Freestyle project

Enter an item name

» Required field



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



Pipeline

Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



External Job

This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

if you want to create a new item from other existing, you can use this option:



Copy from

OK

Adding build step:

Build

Add build step ▾

- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Comments Remover
- Invoke top-level Maven targets
- Run with timeout

Build

Invoke Comments Remover X

Filename ?

Please set a filename

Language ?

Output directory ?

Add build step ▾

This part also has "help" section:

Build

Invoke Comments Remover X

Filename ?

Name of the input file to have comments removed from (from [Comments Remover plugin](#))

Language ?

Language of the input file - helps locate comments (from [Comments Remover plugin](#))

Output directory ?

After execution the result will be in this directory in the workspace (from [Comments Remover plugin](#))

Add build step ▾

Best build Actions

Example of build output and result:

Console Output

```
Started by user admin
Building in workspace C:\Users\Daniel\.jenkins\workspace\test-job
Invoking Comments Remover for filename: input and language: Bash
Installing pip requirements [pip install -r C:\Users\Daniel\.jenkins\comments_remover\requirements.txt -q]...
Executing script [python C:\Users\Daniel\.jenkins\comments_remover\comments_remover.py C:\Users\Daniel\.jenkins\workspace\test-job\input Bash
C:\Users\Daniel\.jenkins\workspace\test-job\comment_remover_output]...
Comments Remover finished processing. Output saved to directory: C:\Users\Daniel\.jenkins\workspace\test-job\comment_remover_output
Finished: SUCCESS
```

| Nazwa | Data modyfikacji | Typ | Rozmiar |
|------------------------|------------------|---------------|---------|
| comment_remover_output | 2017-09-12 20:06 | Folder plików | |
| input | 2017-09-12 19:01 | Plik | 3 KB |

| Nazwa | Data modyfikacji | Typ | Rozmiar |
|----------|------------------|------------|---------|
| rc.input | 2017-09-12 20:06 | Plik INPUT | 1 KB |

Using as a shell/batch command script

Alternatively, user can create plain script build step (Execute shell or Execute Windows batch command) and run comments_remover.py Python script manually.

Build

Execute shell

Command `python3.6 $JENKINS_HOME/comments_remover/comments_remover.py input.cs CSharp output`

See [the list of available environment variables](#)

Advanced...

Add build step

Comments Remover script is unpacked to JENKINS_HOME directory (which is available in script as environmental variable)

Jenkins

Jenkins > test > #8

- Back to Project
- Status
- Changes
- Console Output**
- View as plain text
- Edit Build Information
- Delete Build
- Previous Build

Console Output

Started by user [admin](#)

Building in workspace /home/daniel/.jenkins/workspace/test

```
[test] $ /bin/sh -xe /home/daniel/apache-tomcat-7.0.81/temp/jenkins2764689946492196240.sh
+ python3.6 /home/daniel/.jenkins/comments_remover/comments_remover.py input.cs CSharp output
Finished: SUCCESS
```

If you use Jenkins Struct Plugin, this plugin has a shorthand defined: `commentsremover`.

For contributors

Updating plugin with a new version of Comments Remover

Create ZIP archive of all files required to run Comments Remover - in particular, `comments_remover.py` and `requirements.txt`. Put it in `src/main/resources`. The plugin expects to find `comment_remover.py` and `requirement.txt` on the top lever of the archive.

Debug

```
mvnDebug hpi:run
```

You can attach Java Debugger to a local Java process of Jenkins.

Prepare plugin for distribution

```
mvn package
```

The *.hpi file will be created in target/ directory. To install manually on local Jenkins, copy it to \$JENKINS_HOME/plugins directory.

Releasing new version to Jenkins Update Center

```
mvn release:prepare release:perform
```

Follow the instructions to assign new version number.

You must have an account on jenkins-ci.org with commit rights, and its credentials configured in maven settings:

```
<server>
  <id>maven.jenkins-ci.org</id>
  <username>...</username>
  <password>...</password>
</server>
```

More info here:

<https://wiki.jenkins.io/display/JENKINS/Hosting+Plugins>

For contributors

Updating plugin with a new version of Comments Remover

Create ZIP archive of all files required to run Comments Remover - in particular, `comments_remover.py` and `requirements.txt`. Put it in `src/main/resources`. The plugin expects to find `comment_remover.py` and `requirement.txt` on the top level of the archive.

Debug

```
mvnDebug hpi:run
```

You can attach Java Debugger to a local Java process of Jenkins.

Prepare plugin for distribution

```
mvn package
```

The *.hpi file will be created in target/ directory. To install manually on local Jenkins, copy it to \$JENKINS_HOME/plugins directory.

Releasing new version to Jenkins Update Center

```
mvn release:prepare release:perform
```

Follow the instructions to assign new version number.

You must have an account on jenkins-ci.org with commit rights, and its credentials configured in maven settings:

```
<server>
  <id>maven.jenkins-ci.org</id>
  <username>...</username>
  <password>...</password>
</server>
```

More info here:

<https://wiki.jenkins.io/display/JENKINS/Hosting+Plugins>

For contributors

Updating plugin with a new version of Comments Remover

Create ZIP archive of all files required to run Comments Remover - in particular, `comments_remover.py` and `requirements.txt`. Put it in `src/main/resources`. The plugin expects to find `comment_remover.py` and `requirement.txt` on the top lever of the archive.

Debug

```
mvnDebug hpi:run
```

You can attach Java Debugger to a local Java process of Jenkins.

Prepare plugin for distribution

```
mvn package
```

The `*.hpi` file will be created in `target/` directory. To install manually on local Jenkins, copy it to `$JENKINS_HOME/plugins` directory.

Releasing new version to Jenkins Update Center

```
mvn release:prepare release:perform
```

Follow the instructions to assign new version number.

You must have an account on jenkins-ci.org with commit rights, and its credentials configured in maven settings:

```
<server>
  <id>maven.jenkins-ci.org</id>
  <username>...</username>
  <password>...</password>
</server>
```

More info here:

<https://wiki.jenkins.io/display/JENKINS/Hosting+Plugins>